# Cheat Sheet: In-depth Understanding of Advanced React Functionality

| Hooks and form management | Description | Code example |
|---|---|---|
| **useState()** | `useState()` hook can manage states of the React function component where you can declare any data type, for example, boolean, object, array, string. | `import React, { useState, useEffect } from 'react';`<br>`function SideEffect() {`<br>`  const [empId, setEmpId] = useState(100);`<br>`  return (`<br>`  <div>`<br>`  <p>{empId}</p>`<br>`  </div>`<br>`  );`<br>`}`<br>`export default SideEffect;` |
| **useEffect()** | `useEffect` is a React hook that allows you to perform side effects in functional components. A side effect refers to any operation that you need to execute as soon as the page loads without calling those operations/functionalities separately, such as fetching data from an API. | `import React, { useState, useEffect } from 'react';`<br>`function SideEffect() {`<br>`  const [foods, setFoods] = useState([]);`<br>`  useEffect(() => {`<br>`  fetch('https://api.npoint.io/d542b9ad99f501ab3dbf')`<br>`  .then(response => response.json())`<br>`  .then(data => {`<br>`  console.log(data);`<br>`  setFoods(data);`<br>`  })`<br>`  .catch(error => console.error('Error fetching users:', error));`<br>`  },[]); // Empty dependency array means this effect runs only once when the component mounts`<br>`  return (`<br>`  <div>`<br>`  <h1>Food List</h1>`<br>`  <ul>`<br>`  {foods.map((food)=>{`<br>`  return (<>`<br>`  <li><h1>{food.name}</h1></li>`<br>`  <p>food.description</p>`<br>`  <p>food.price</p>`<br>`  <p>food.category</p>`<br>`  <p>food.ingredients</p>`<br>`  <img src={food.image_url} alt="" height='100px' width='100px' />`<br>`  </>`<br>`  )`<br>`  })}`<br>`  </ul>`<br>`  </div>`<br>`  );`<br>`}`<br>`export default SideEffect;` |
| **Custom hook** | You can use custom hooks in any other component. In this code snippet, there is one function component known as `UseToggle`, which serves as a custom hook, and another function component `ToggleButton`, which will use this custom hook. | `//ToggleButton`<br>`import { useState } from 'react';`<br>`import UseToggle from './UseToggle';`<br>`function ToggleButton() {`<br>`  const [isToggled, toggle] = UseToggle(false);`<br>`  return (`<br>`  <div>`<br>`  <h1>Toggle Button</h1>`<br>`  <button onClick={toggle}>`<br>`  {isToggled ? 'ON' : 'OFF'}`<br>`  </button>`<br>`  </div>`<br>`  );`<br>`}`<br>`export default ToggleButton;`<br>`//UseToggle.jsx`<br>`import { useState } from "react";`<br>`function UseToggle(initialValue = false) {`<br>`  const [value, setValue] = useState(initialValue);`<br>`  const toggle = () => {`<br>`  setValue(!value);`<br>`  };`<br>`  return [value, toggle];`<br>`  }`<br>`  export default UseToggle` |

| | | |
|---|---|---|
| | | |
| **fetch api method** | Fetch method can fetch data using API. | `const apiUrl = 'https://jsonplaceholder.typicode.com/posts';`<br>`fetch(apiUrl)`<br>`  .then(response => response.json())`<br>`  .then(data => {`<br>`  console.log(data);`<br>`  })`<br>`  .catch(error => {`<br>`  console.error('There was a problem with the fetch operation:', error);`<br>`  });` |
| **axios api method** | Axios method can fetch data using API. | `import axios from 'axios';`<br>`const apiUrl = 'https://jsonplaceholder.typicode.com/posts';`<br>`axios.get(apiUrl)`<br>`  .then(response => {`<br>`  console.log(response.data);`<br>`  })`<br>`  .catch(error => {`<br>`  console.error('There was a problem with the fetch operation:', error);`<br>`  });` |
| **onChange** | The `onChange` event attribute is often used in HTML and React to track when the value of an input field changes, like a text input. The onChange event occurs when a user writes something into an input field. This attribute lets you record and handle the changes. | `import React, { useState } from 'react';`<br>`function FormData() {`<br>`  const [empName, setEmpName] = useState('');`<br>`  const handleChange = event => {`<br>`  setEmpName(event.target.value);`<br>`  };`<br>`  const handleSubmit = event => {`<br>`  event.preventDefault();`<br>`  console.log('Form submitted:', empName);`<br>`  };`<br>`  return (`<br>`  <div>`<br>`  <h2>My Form</h2>`<br>`  <form onSubmit={handleSubmit}>`<br>`  <label>`<br>`  Input:`<br>`  <input type="text" value={empName} onChange={handleChange} />`<br>`  </label>`<br>`  <button type="submit">Submit</button>`<br>`  </form>`<br>`  </div>`<br>`  );`<br>`}`<br>`export default FormData;` |
| **Redux toolkit** | Redux toolkit can be installed using `npm` | `npm install @reduxjs/toolkit.` |